AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings of claims in the application:

LISTING OF CLAIMS

1-42 (Cancelled)

43. (Currently Amended) A small footprint device comprising:

at least one processing element configured to execute groups of one or more program modules in separate contexts, said one or more program modules comprising zero or more sets of executable instructions and zero or more sets of data definitions, said zero or more sets of executable instructions and said zero or more data definitions grouped as object definitions, each context comprising a protected object instance space such that at least one of said object definitions is instantiated in association with a particular context;

a context barrier for separating and isolating said contexts, said context barrier configured for controlling execution of at least one instruction of one of said zero or more sets of instructions comprised by a program module based at least in part on whether said at least one instruction is executed for an object instance associated with a first one of said one or more separate contexts and whether said at least one instruction is requesting access to an instance of an object definition associated with a second one of said one or more separate contexts, said context barrier further configured to prevent said access if said access is unauthorized and enable said access if said access is authorized; and

a global data structure for permitting one program module to access information from another program module by bypassing said context barrier, ~~The small footprint device of claim 1~~ wherein an object instance is associated with a context by recording the name of said context in a header of said object instance, information in said header inaccessible to said one or more program modules.

44-45 (Cancelled)

46. (Currently Amended) A method of operating a small footprint device that includes a processing machine, wherein program modules are executed on the processing machine, the method comprising:

separating contexts using a context barrier, said context barrier configured to for controlling execution of at least one instruction of one of said zero or more sets of instructions comprised by a program module based at least in part on whether said at least one instruction is executed for an object instance associated with a first one of said one or more separate contexts and whether said at least one instruction is requesting access to an instance of an object definition associated with a second one of said one or more separate contexts, said separating further comprising:

preventing said access if said access is unauthorized; and

enabling said access if said access is authorized;

executing groups of one or more program modules in separate contexts, said one or more program modules comprising zero or more sets of executable instructions

and zero or more sets of data definitions, said zero or more sets of executable instructions and said zero or more data definitions grouped as object definitions, each context comprising a protected object instance space such that at least one of said object definitions is instantiated in association with a particular context; and

permitting access to information across said context barrier by bypassing said context barrier using a global data structure, ~~The method of claim 32~~ wherein an object instance is associated with a context by recording the name of said context in a header of said object instance, information in said header inaccessible to said one or more program modules.

47-48 (Cancelled)

49. (Currently Amended) A method of permitting access to information on a small footprint device from a first program module to a second program module separated by a context barrier, said small footprint device comprising:

at least one processing element configured to execute groups of one or more program modules in separate contexts, said one or more program modules comprising zero or more sets of executable instructions and zero or more sets of data definitions, said zero or more sets of executable instructions and said zero or more data definitions grouped as object definitions, each context comprising a protected object instance space such that at least one of said object definitions is instantiated in association with a particular context ;

a memory comprising instances of objects; and

a context barrier for separating and isolating said contexts, said context barrier configured for controlling execution of at least one instruction of one of said zero or more sets of instructions comprised by a program module based at least in part on whether said at least one instruction is executed for an object instance associated with a first one of said one or more separate contexts and whether said at least one instruction is requesting access to an instance of an object definition associated with a second one of said one or more separate contexts, said context barrier further configured to prevent said access if said access is unauthorized and enable said access if said access is authorized, the method comprising:

creating a global data structure which may be accessed by at least two program modules; and

using said global data structure to permit access to information across said context barrier by bypassing said context barrier, ~~The method of claim 34~~ wherein an object instance is associated with a context by recording the name of said context in a header of said object instance, information in said header inaccessible to said one or more program modules.

50-51 (Cancelled)

52. (Currently Amended) A method of communicating across a context barrier separating program modules on a small footprint device, said small footprint device comprising:

at least one processing element configured to execute groups of one or more program modules in separate contexts, said one or more program modules comprising zero or more sets of executable instructions and zero or more sets of data definitions, said zero or more sets of executable instructions and said zero or more data definitions grouped as object definitions, each context comprising a protected object instance space such that at least one of said object definitions is instantiated in association with a particular context;

a memory comprising instances of objects; and

a context barrier for separating and isolating said contexts, said context barrier configured for controlling execution of at least one instruction of one of said zero or more sets of instructions comprised by a program module based at least in part on whether said at least one instruction is executed for an object instance associated with a first one of said one or more separate contexts and whether said at least one instruction is requesting access to an instance of an object definition associated with a second one of said one or more separate contexts, said context barrier further configured to prevent said access if said access is unauthorized and enable said access if said access is authorized, the method comprising:

creating a global data structure;

permitting at least one program module to write information to said global data structure; and

having at least one other program module read information from said global data structure, bypassing said context barrier, ~~The method of claim 35~~ wherein an object instance is associated with a context by recording the name of said

context in a header of said object instance, information in said header inaccessible to said one or more program modules.